FIG.1
(PRIOR ART)

MICROPROCESSOR 40

PROCESSOR NODE

BRIDGE 41

SYSTEM MEMORY 42

SYSTEM BUS (PCI OR OTHER) 43

OTHER PERIPHERAL 47

NETWORK INTERFACE 44

VIDEO CARD 45

DISK/STORAGE DEVICE 46

FIG.2

PACKET NODE A 48a

FUNCTIONAL UNIT 51a
(PROCESSOR, MEM. DEVICES, STORAGE DEVICES, I/O DEVICES, DISPLAY DEVICES ETC.)

LINK INTERFACE 49a

50

PACKET NODE B 48b

LINK INTERFACE 49b

FUNCTIONAL UNIT 51b

FIG. 3

54 ⟶ | HEADER | } 52

55 ⟶ BODY OR DATA
(0-256 BYTES)

53 ⟶

WIDTH = 2 BYTES

SIZE = 2 BYTES TO A
MAXIMUM OF
258 BYTES

FIG. 4

56   52 ⟶   58

54   57   59

FIG. 5
IDLE PACKET

MSB   55a ⟶   LSB

| HEADER | } 52

54

SIZE: MIN. 2 BYTES
NO DATA

55b

MSB   LSB

54 ⟩ HEADER ⟩ 52

BODY

53

FIG. 6
REQUEST PACKET

MIN. SIZE :   4 BYTES
MAX. SIZE : 258 BYTES

55c

MSB   LSB

54 ⟩ HEADER ⟩ 52
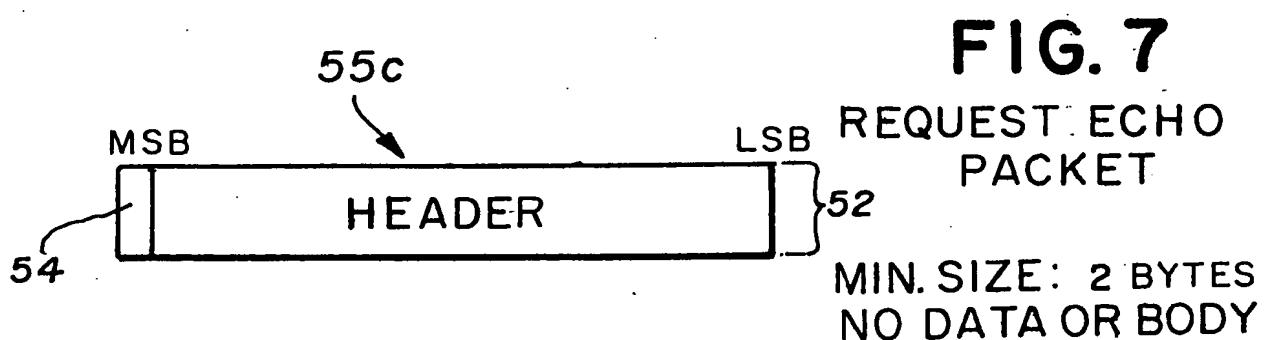
FIG. 7
REQUEST ECHO PACKET

MIN. SIZE : 2 BYTES
NO DATA OR BODY

MSB   LSB

54 ⟩ HEADER ⟩ 52

BODY OR DATA
(BODY OF RESPONSE)

55d

53
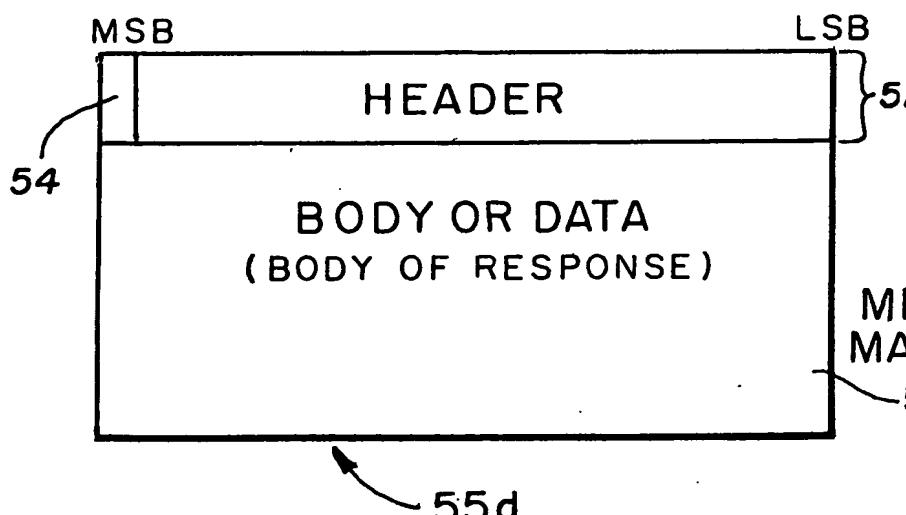
FIG. 8
RESPONSE PACKET

MIN. SIZE : 4 BYTES
MAX. SIZE : 258 BYTES

FIG.9

FUNCTIONAL DIAGRAM OF LINK INTERFACE

ONE WAY LINK

PACKET COMMUNICATION

ONE WAY LINK

PHYSICAL LINC

50

49

REQUEST QUEUE 64

ECHO WAITING QUEUE 65

STORE ACCUMULATOR 61

RESPONSE QUEUE 66

LINC CACHE 63

CONTROL 62

HOST INTERFACE 60

DATA 69

ADDR 68

CNTRL 70

48 HOST: μp or MEM.or DISK or OTHER STORAGE DEVICE or I/oDEVICE or VIDEO DEVICE or NETWORK DEVICE etc.
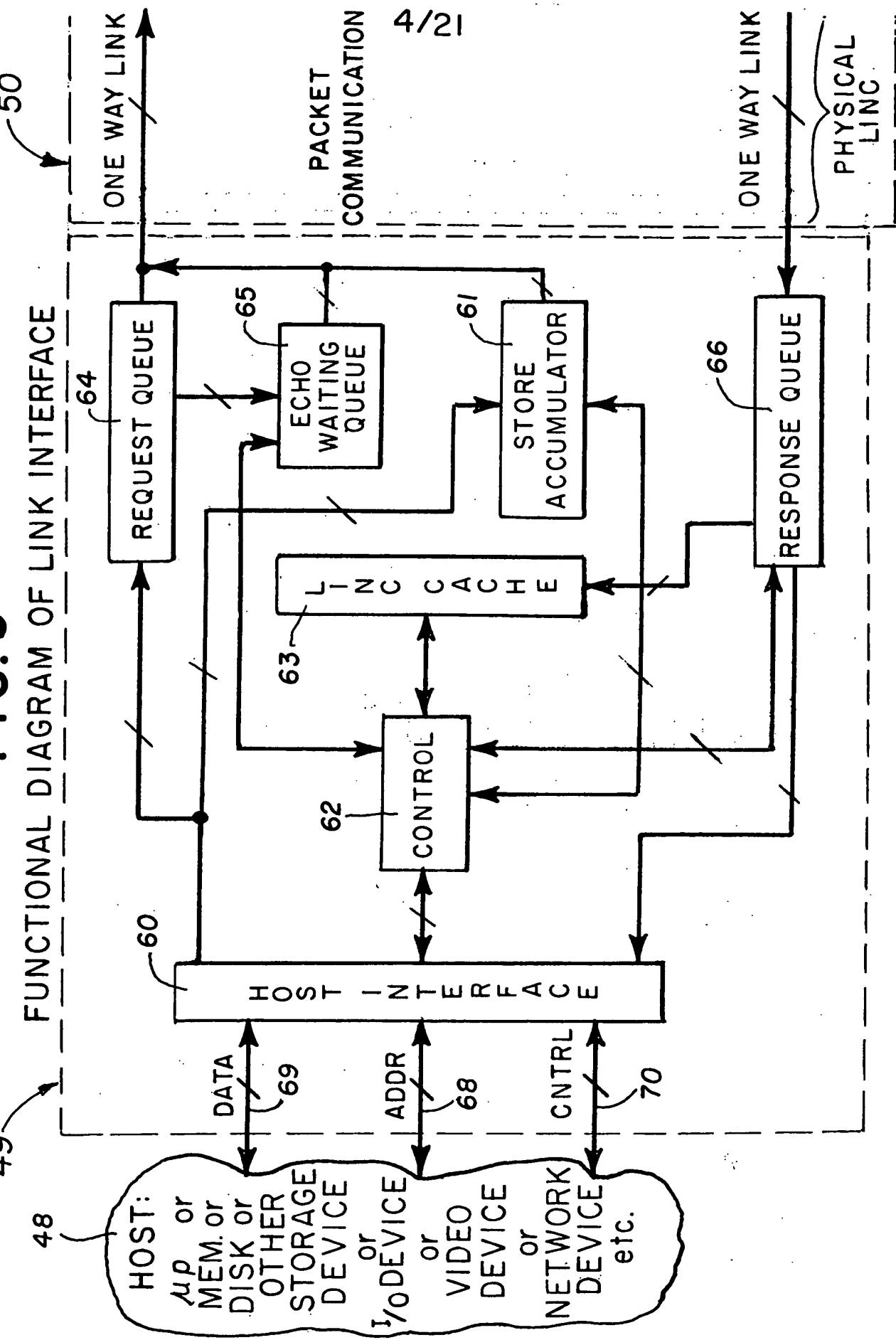
FIG.10a

POSSIBLE REQUEST QUEUE STRUCTURES

WIDTH=2 BYTES +1 BIT FOR FRAME (67)

FROM PACKETIZE →

3rd.PACKET 2nd.PACKET 1st.PACKET

H E A D E R 55	D A T A P A C K E T	H E A D E R 55	D A T A P A C K E T

52 64 TO TRANSMIT →

53 52

SET SET

NOT SET

SET

53

NOT SET

SET

HEADER

NOT SET

DATA PACKET

53

67 FRAME BIT
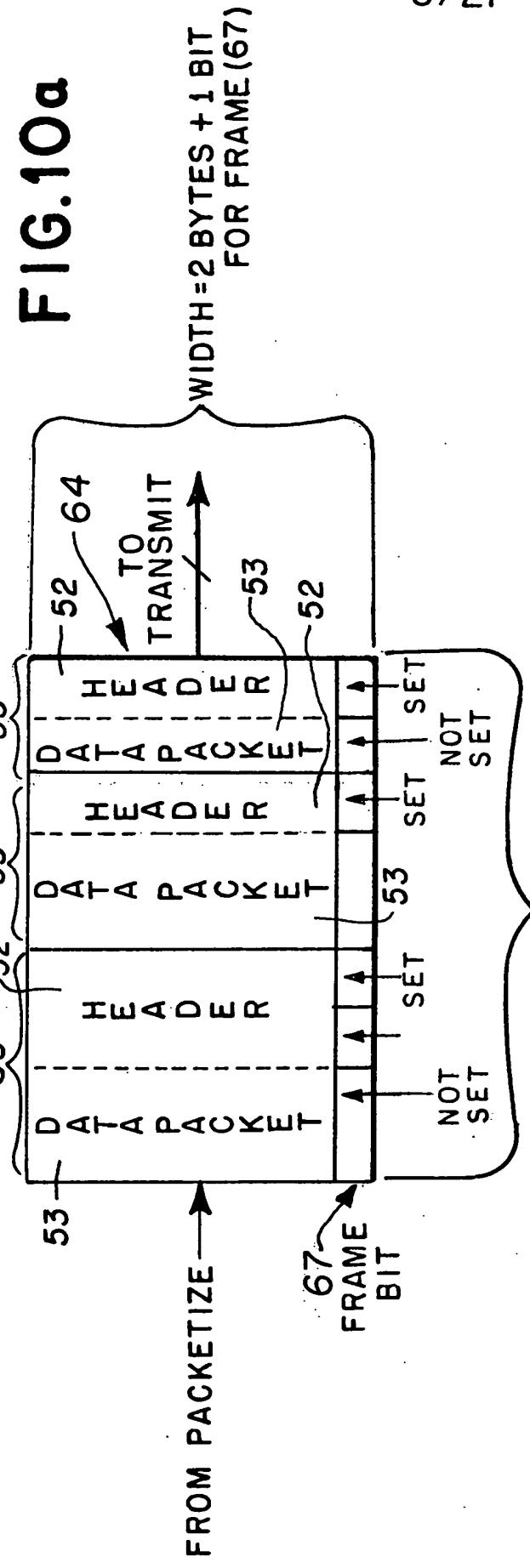
DEPTH = **n** x MAXIMUM PACKET SIZE (258 BYTES) n=1,2,3...
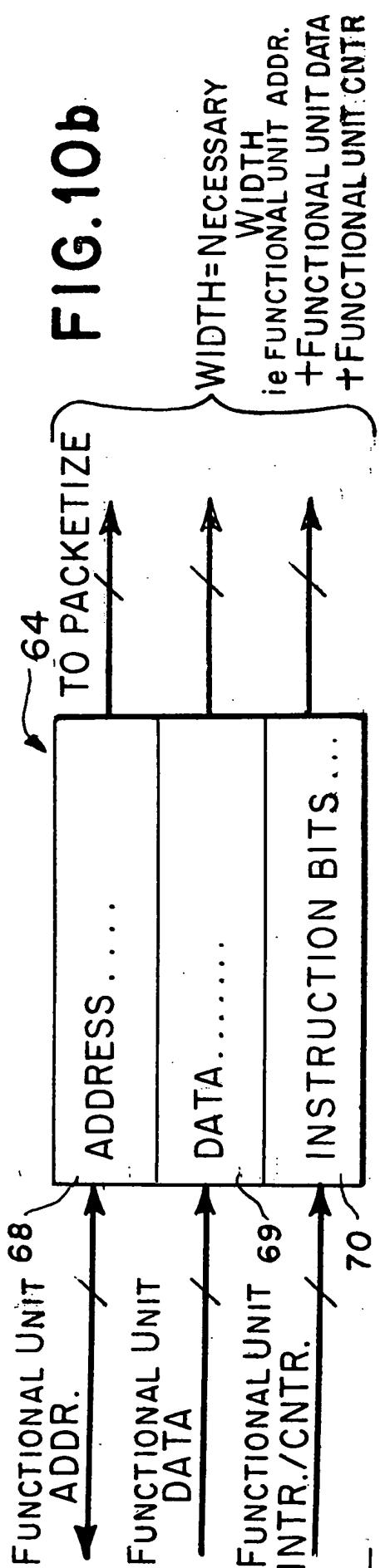
FIG.10b

WIDTH=NECESSARY WIDTH
ie FUNCTIONAL UNIT ADDR.
+FUNCTIONAL UNIT DATA
+FUNCTIONAL UNIT CNTR

64 TO PACKETIZE

ADDRESS.......
DATA.......
INSTRUCTION BITS.....

FUNCTIONAL UNIT 68 ADDR.

FUNCTIONAL UNIT DATA

FUNCTIONAL UNIT 69 INTR./CNTR.

70

POSSIBLE RESPONSE QUEUE STRUCTURES
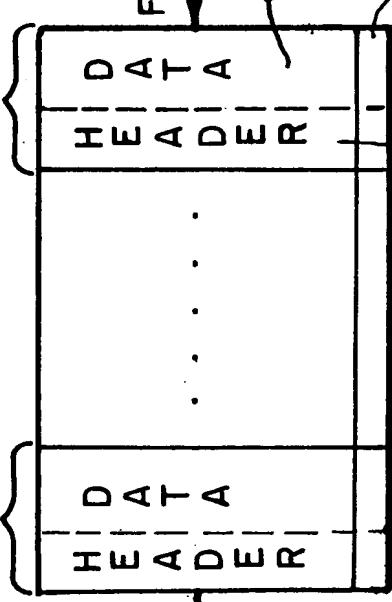
FIG.11a

WIDTH = 2 BYTES +1 BYTE FOR FRAME

FROM RECEIVE

FRAME BIT — 67

PACKET 55

| H E A D E R | D A T A |
| | |

53

52

PACKET 55

| H E A D E R | D A T A |

66

TO DECODE

FIG.11b

AFTER DECODE

WIDTH=NECESSARY WIDTH

70

| CNTRL . . . . . | DATA . . . . | ADDRESS . . . . |

69

68

66

TO FUNCTIONAL UNIT INTERFACE

FIG.12
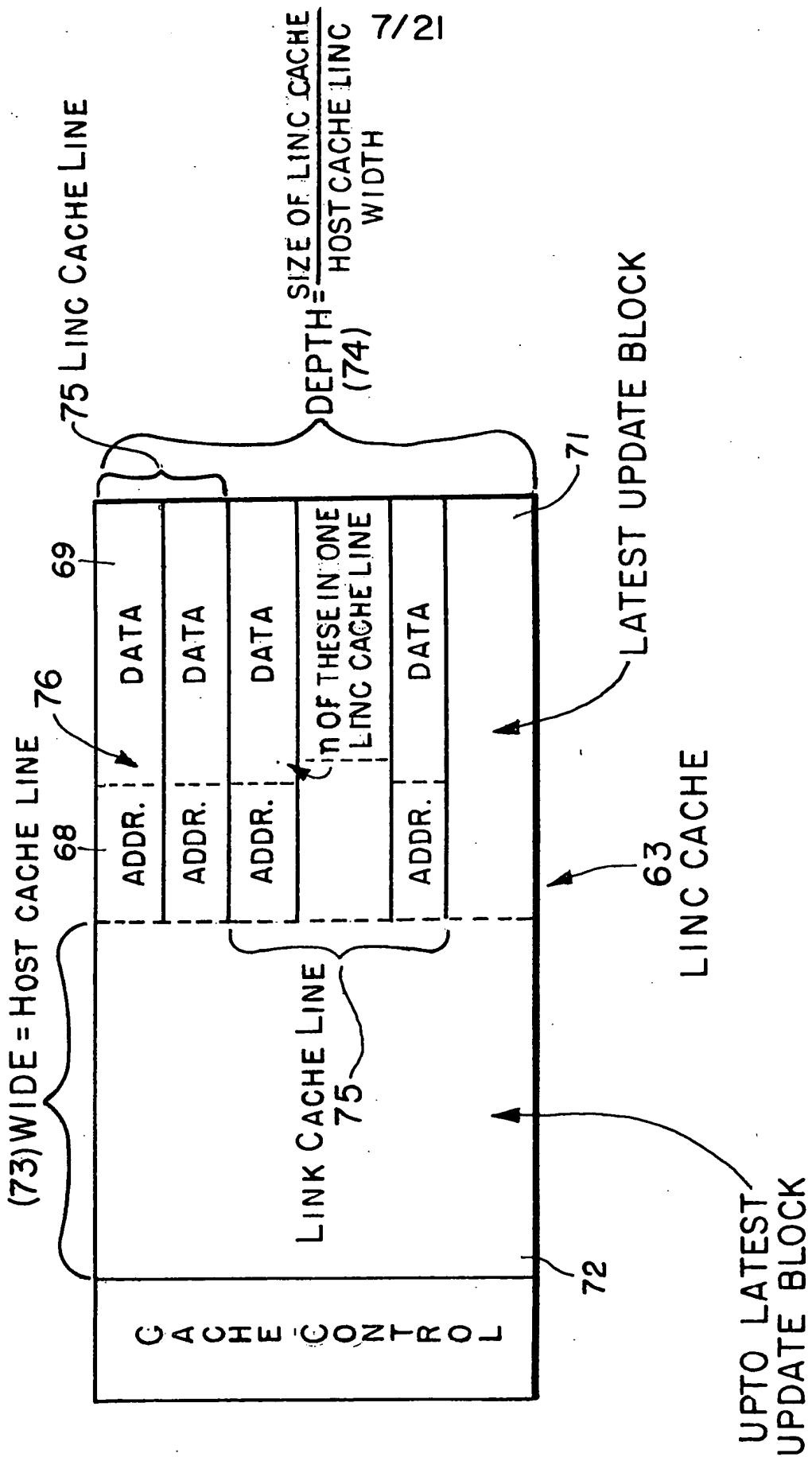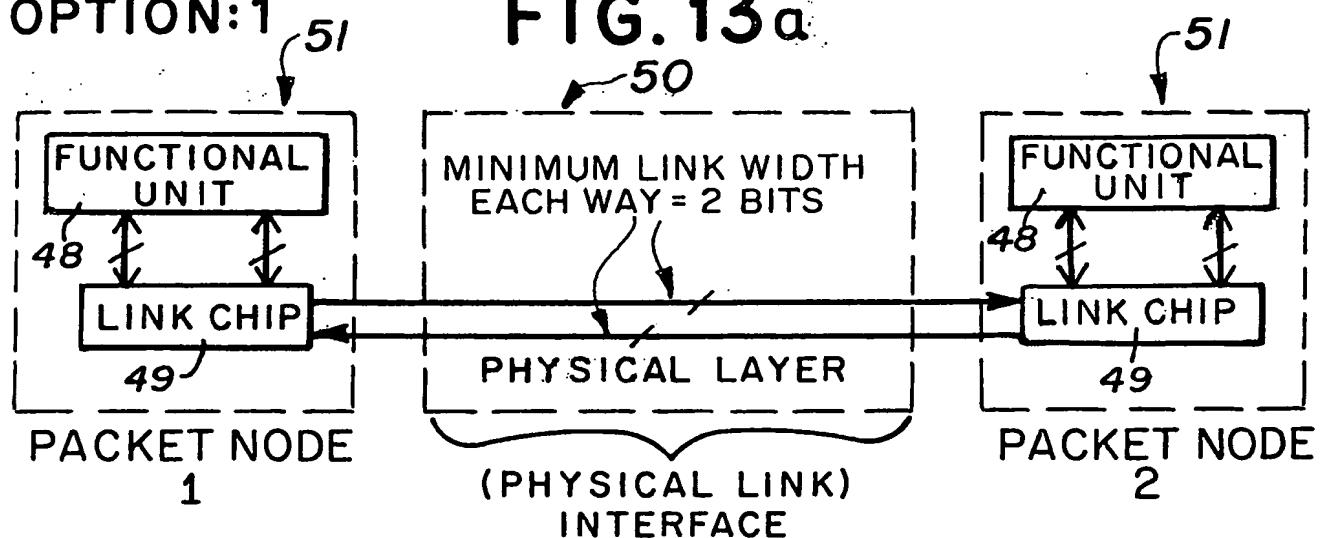
FIG. 13a

OPTION: 1

A SAMPLE TWO NODE LINK WHERE NODE 1
FUNCTIONAL UNIT IS THE PROCESSOR AND
NODE 2 FUNCTIONAL UNIT IS THE MEMORY.

FIG. 13b

OPTION: 2

SYSTEM MEMORY

48

MEMORY CONTROLLER 21b

LINK INTERFACE 49

MINIMUM LINK WIDTH =2 BITS

50

INTERFACE 49

LINK 51

PROCESSOR CORE (ALU, FPU) 48

FIG.14

FIG.15

IDLE PACKET

MSB

54

55a   59   LSB   52

56   57   58

HEADER,
NO BODY

ACTIVE
FIELD

FIG.16

REQUEST PACKET:
MSB

54

55b   59   52

56

57   58   53

LSB

HEADER,
(ALL FIELDS
ARE ACTIVE)

FIG.17

REQUEST ECHO PACKET: 57   58   55c   59
MSB

54

56

LSB

52

ONLY ACTIVE FIELDS

FIG.18

RESPONSE PACKET

MSB · Active · 57 · 55d · 58 · 59 · Active

54 · FLOW CONTROL · 56 · 52

BODY OF RESPONSE

53
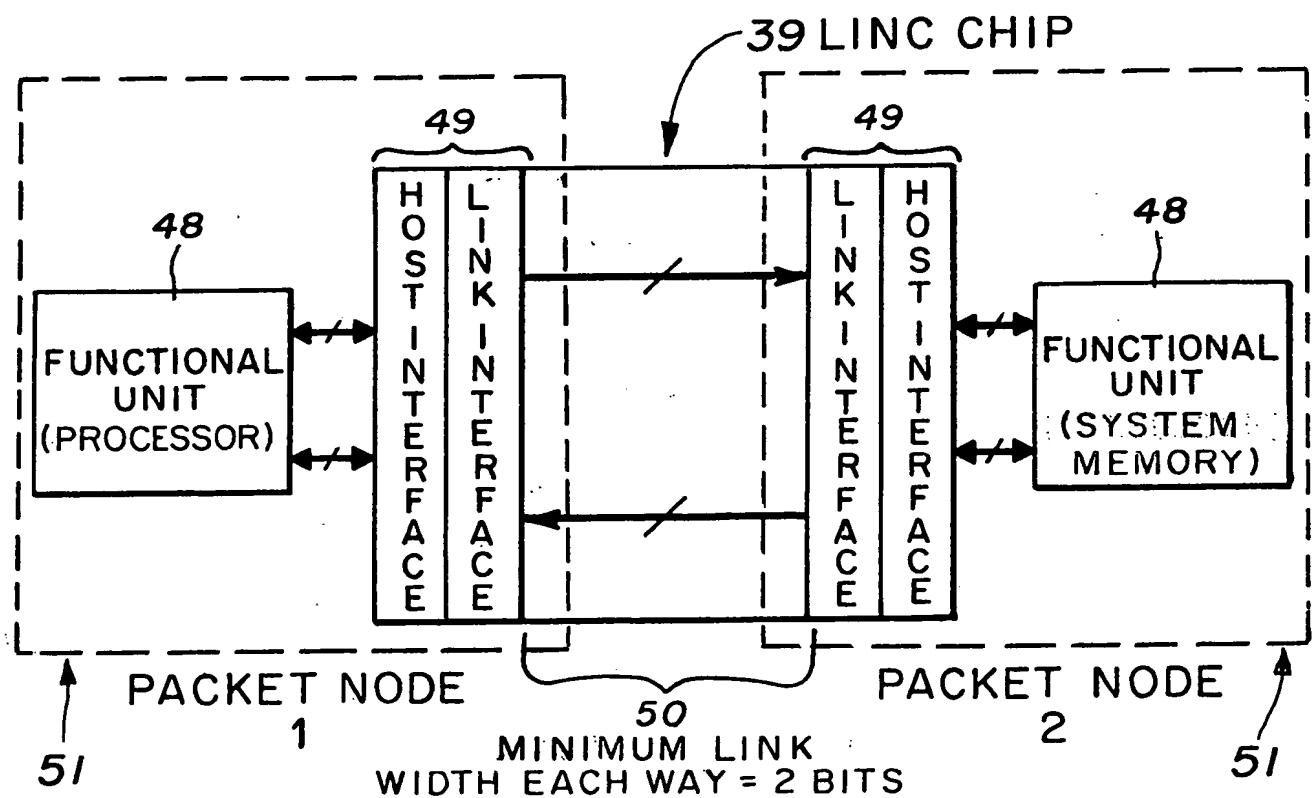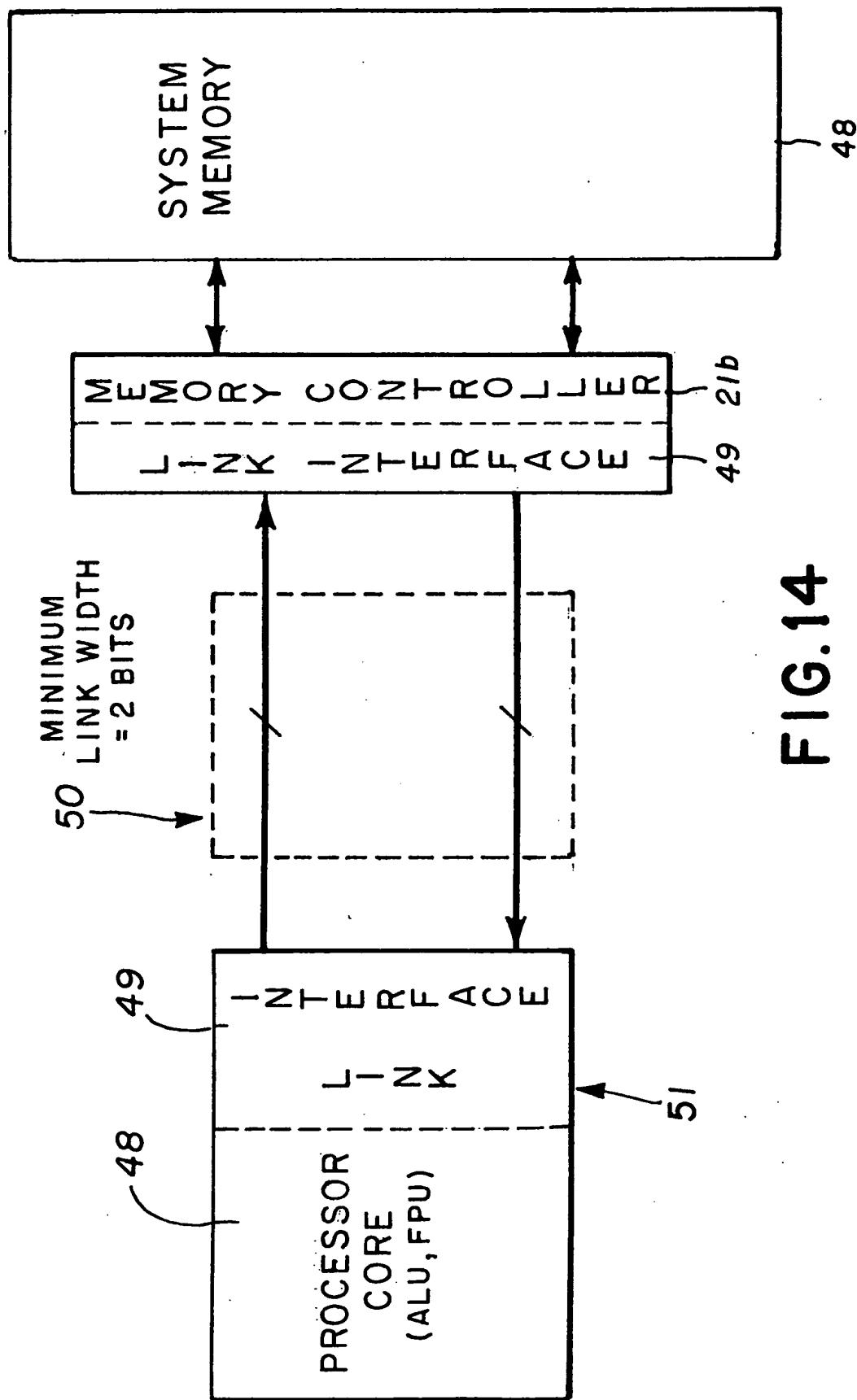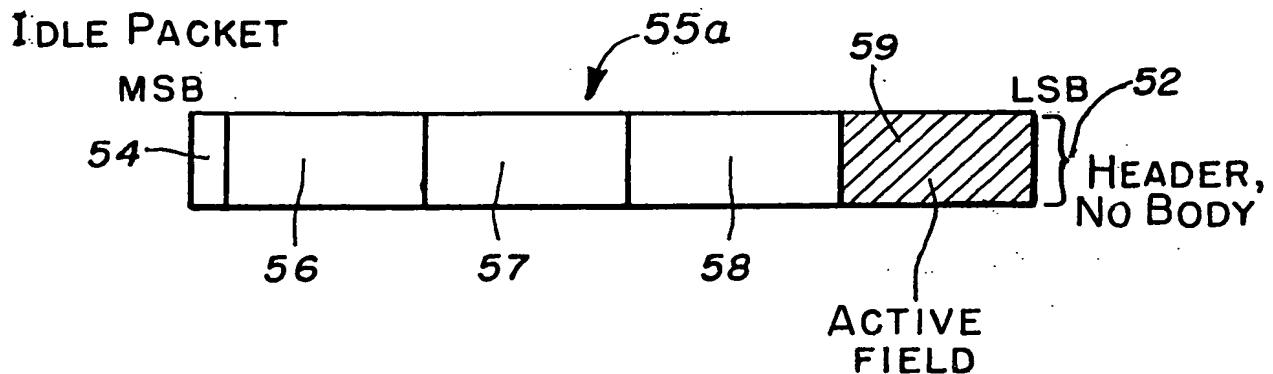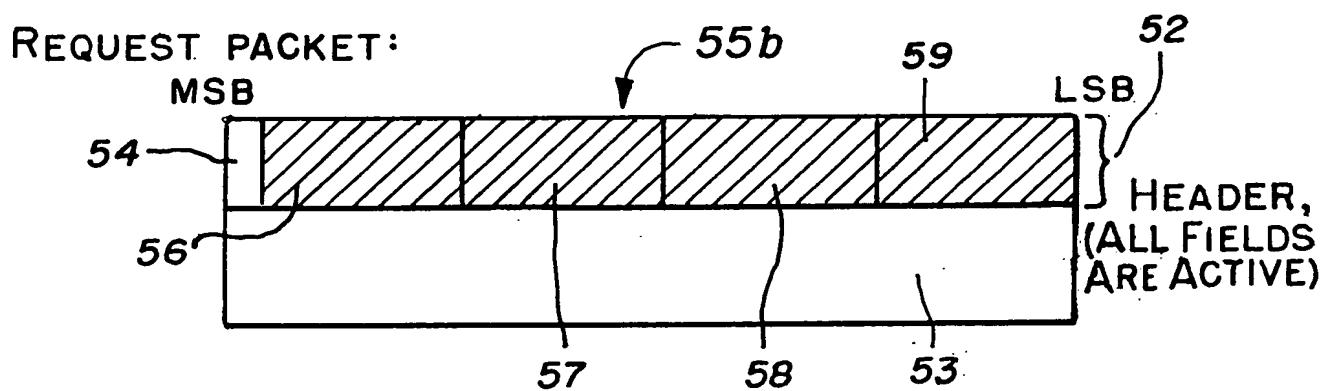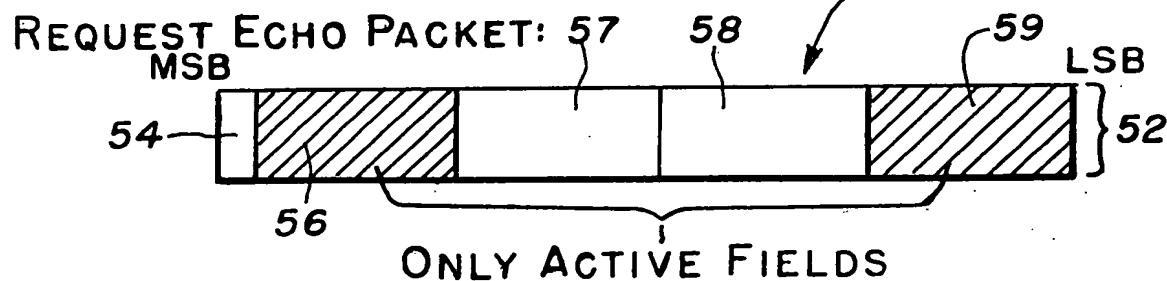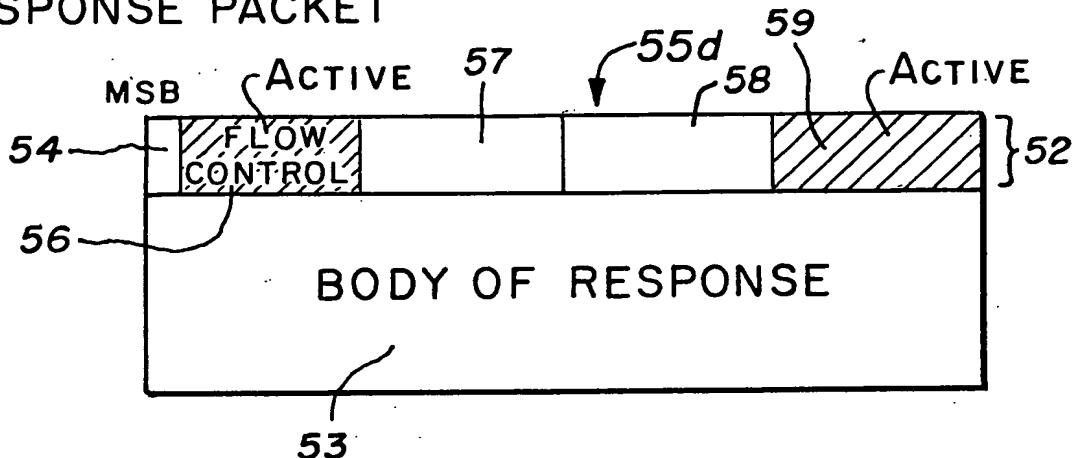
FIG.19

REQUEST PACKET FOR A LOAD

HEADER — 52

55 — ADDRESS TO LOAD FROM — 53

REQUEST PACKET FOR A STORE

55 — HEADER — 52

53 { ADDR. TO STORE TO
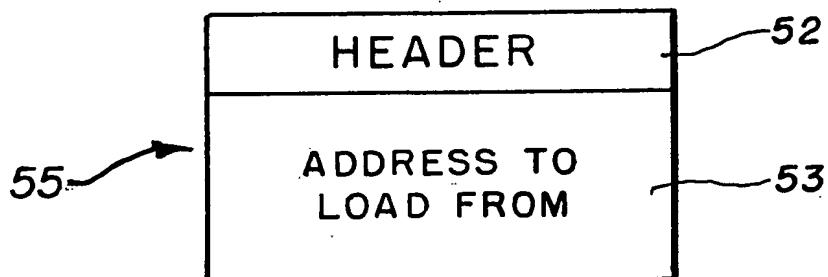
DATA TO STORE
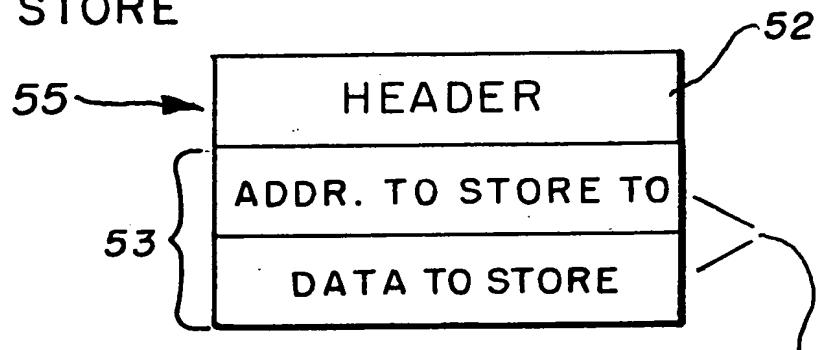
K OF EACH WHERE K IS THE NUMBER OF STORES IN THE STORE ACCUMULATOR.

FIG. 20
HOST CHIP TRANSMIT HALF
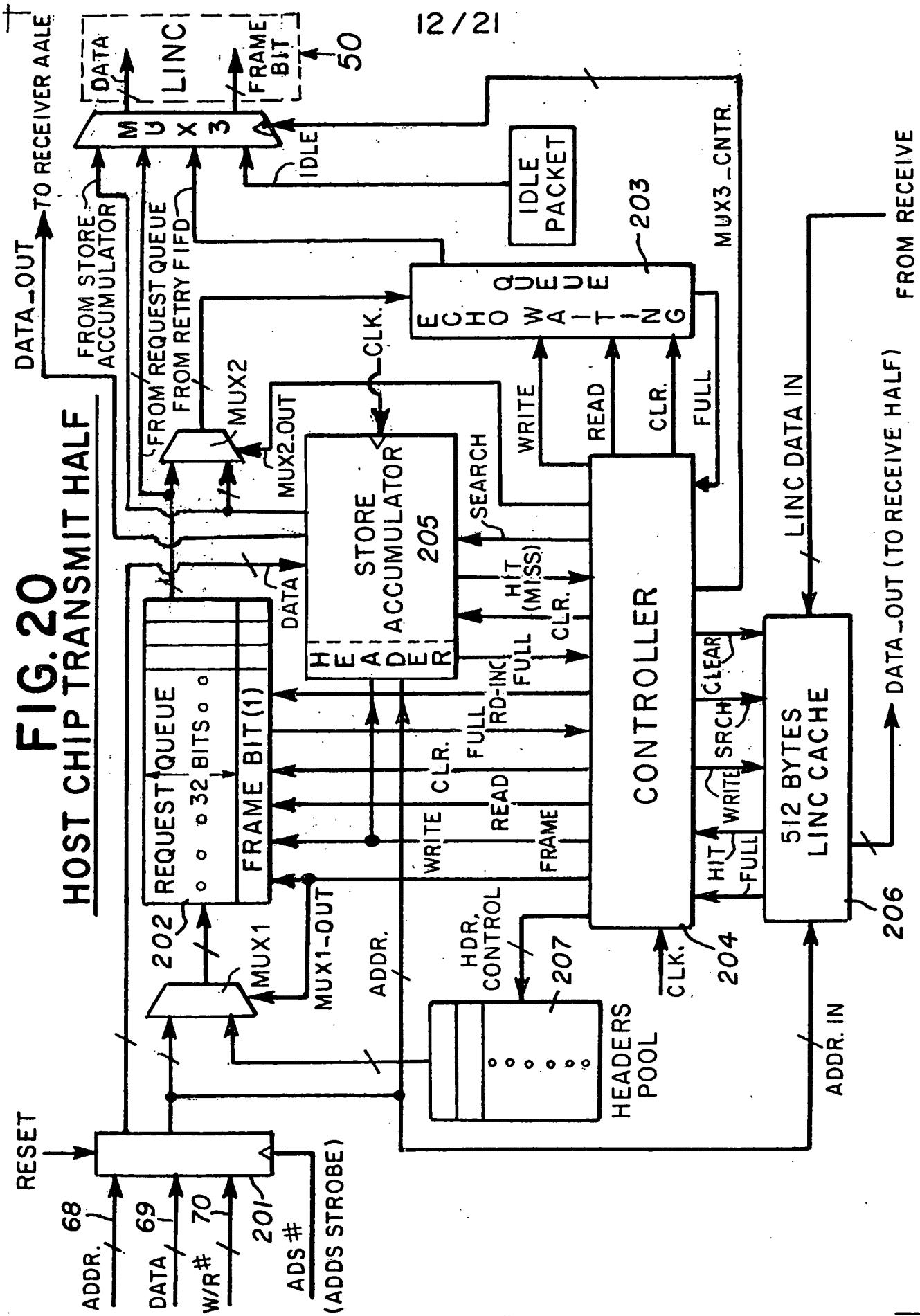
FIG.21
RECEIVE HALF HOST CHIP

13/21

RESETS ARE ALL GLOBAL
CLK=250 MHZ CLOCK
LINC CLK⇒DUAL EDGE TRIGGERED
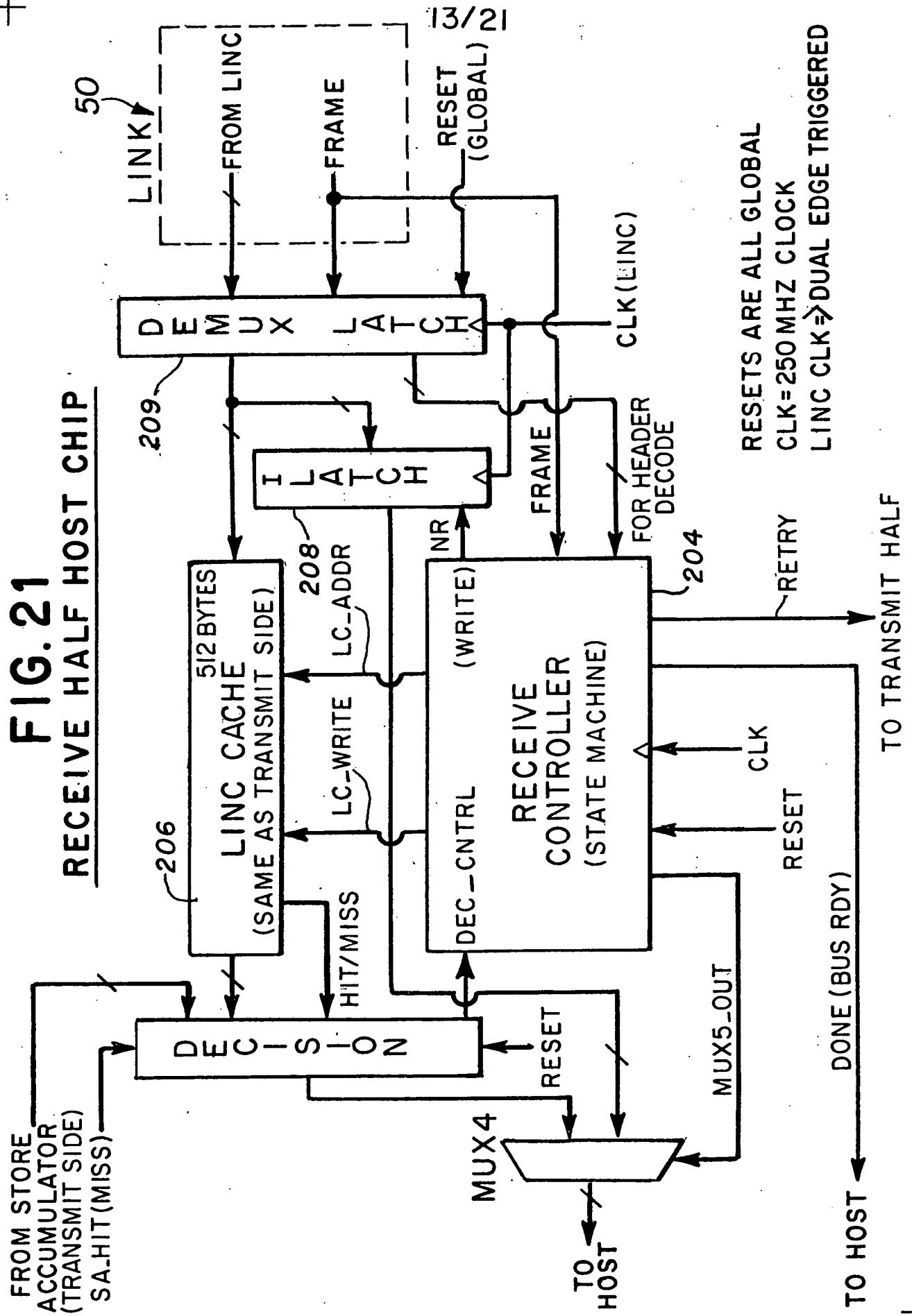
FIG. 22
MEMORY CHIP RECEIVE HALF

CONTROL TO MEMORY

MEMORY CONTROLLER 211

HIT/MISS

DATA

ADDR (TO MUX ON TRANSMIT SIDE)

TO MEMORY

FROM MEMORY

212

512 BYTES

LINC CACHE (SAME FOR TRANSMIT)

ADDR.

DEQUEUE

L/S BIT

REQ QUEUE 210

ADDRESS

DATA

LOAD STORE BIT

SEARCH

AMNT

LOAD STORE BIT

CONTROLLER (STATE MACHINE) 209

HIT (MISS)

TO CONTROLLER ON TRANSMIT HALF

DECISION 208

DATA

CLK

CLK

LINC 50

FRAME

CLK (LINK CLK.)

AMOUNT TO TRANSMIT

FIG. 23
MEMORY CHIP TRANSMIT HALF
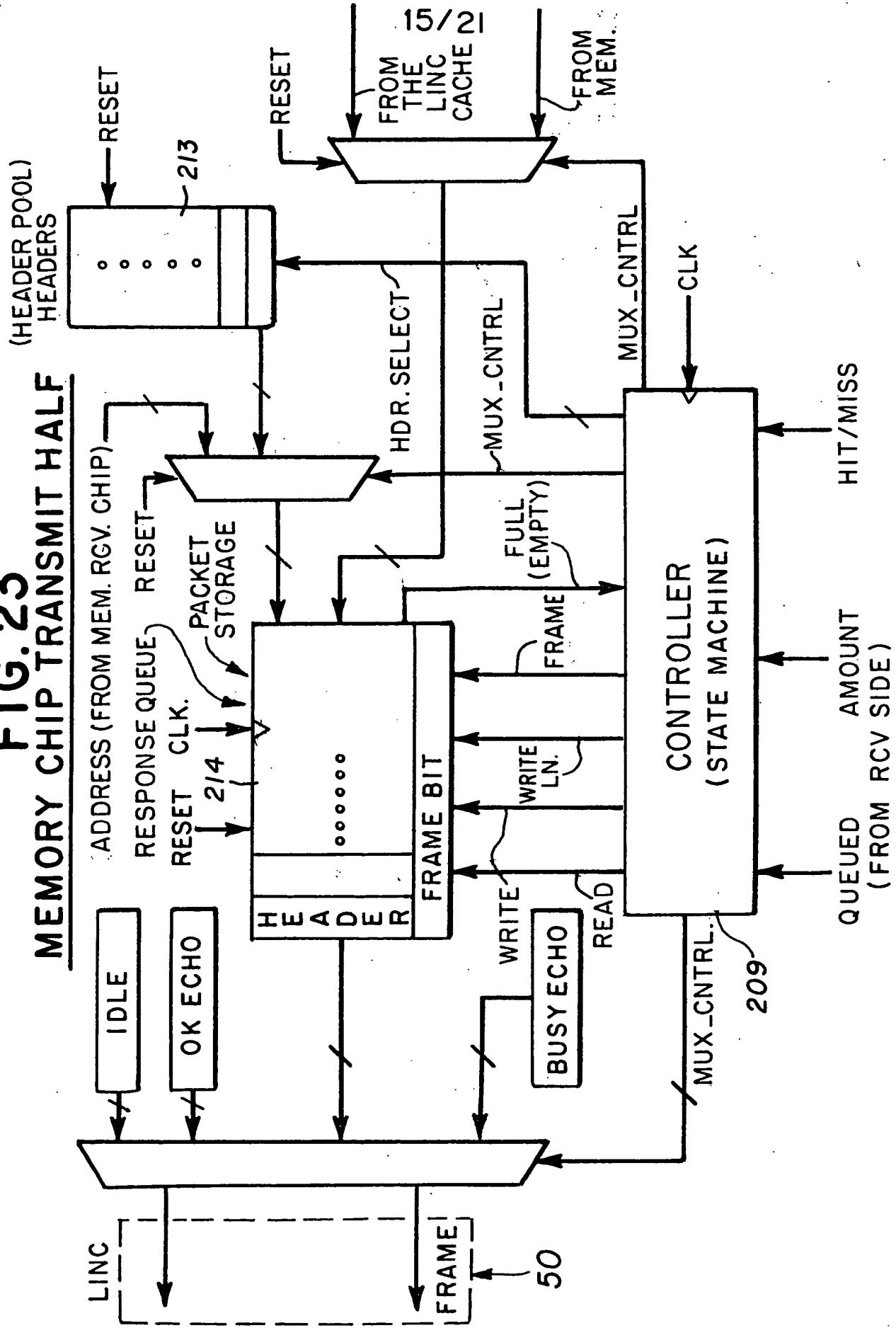
15/21

FIG. 24

PACKET NODE

IT IS POSSIBLE TO
HAVE MORE THAN
4 PACKET NODES
ON A RING.

PACKET
NODE

PACKET
NODE

EXPANSION BUS
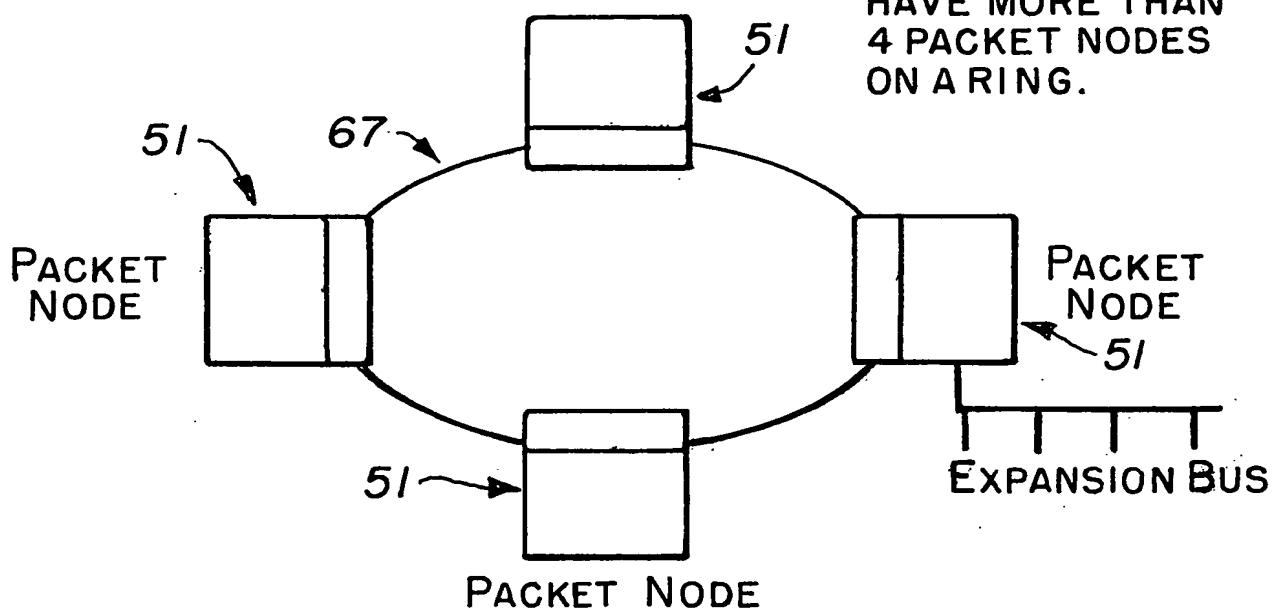
PACKET NODE

FIG. 25

PACKET NODE

PACKET NODE

PACKET NODE

PACKET NODE
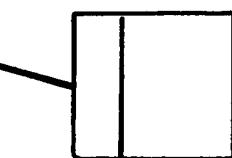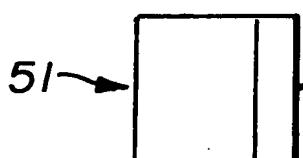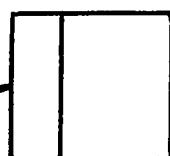
CROSSBAR
OR SWITCH
INTERCONNECT

FIG. 26

POSSIBLE RESPONSE PACKETS FOR A REQUEST WHEN ACCESS TO TLB AND OR BTB IN THE PROCESSOR IS AVAILABLE

55d —

RESPONSE PACKET

REQUEST (MISS/ERROR IN SPEC- ULATION EXECUTION) PACKET

HEADER
DATA FOR ADDRESS n
DATA FOR ADDRESS n +OFFSET
DATA FOR ADDRESS y
DATA FOR ADDRESS n + 2 × OFFSET
DATA FOR ADDRESS z

} 52 {

HEADER
ADDRESS n

55b
LOAD
FROM

53

53 {

FOR PC=X

FOR PC=X+1

FOR PC=X +1 AND JUMP

FOR PC=X+2

CORRESPONDS TO PC (PROGRAM COUNTER) = SOME NUMBER X

DATA NEEDED IN CASE OF A JUMP AT PC=X+2
DATA NEEDED IN CASE OF NO JUMP AT PC=X+2

.
.
.

AND SO ON.

FOR PROGRAMS WITH VERY HIGH LOCALITY (GRAPHICS, MULTIMEDIA ETC.)

RESPONSE PACKET

55d

52 {

HEADER
DATA FOR ADDRESS n
DATA FOR ADDRESS n + OFFSET
DATA FOR ADDRESS n +2 OFFSET
. . . AND SO ON.

FOR PC = X

FOR PC = X+1

FOR PC = X+2

53
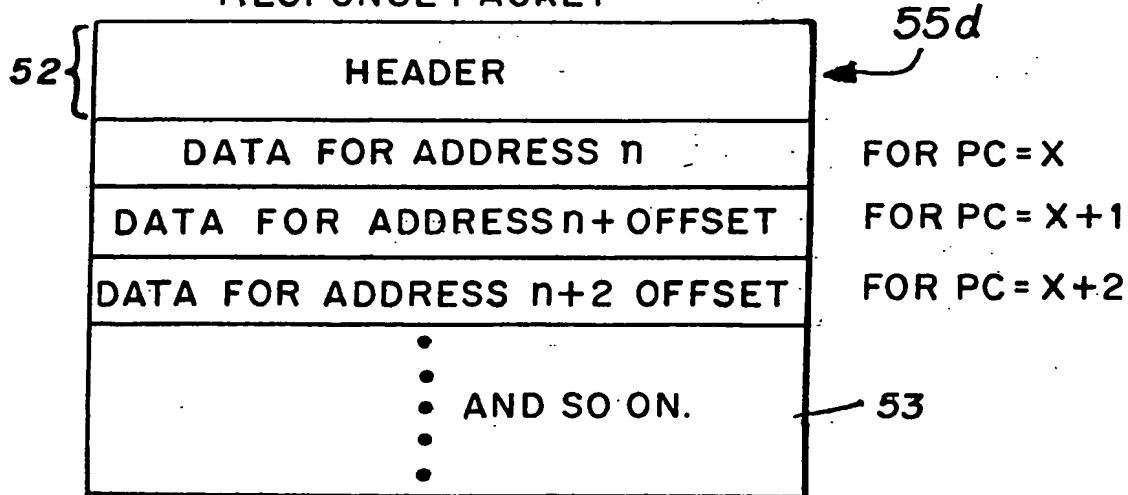
FIG. 27
PROCESSOR NODE RECEIVE PROTOCOL

FOR PRESENT DETAILED IMPLEMENTATION, THE MEMORY
NODE IS EXPECTED TO SEND ONLY THE FOLLOWING PACKETS.

```
          IF REQUEST ECHO
              PACKET
```

```
   IF ECHO = OKAY          ELSE LEAVE REQUEST IN
                           ECHO WAITING QUEUE
```

```
  DEQUEUE REQUEST
    FROM ECHO
   WAITING QUEUE
```

```
        IF RESPONSE PACKET
```

```
           STRIP PACKET
```

```
  FILL LINC      ←— IN PARALLEL —→    PUT N* LINES IN
   CACHE                              RESPONSE QUEUE
```

*LINE HERE INDICATES ONE
PROCESSOR REQUESTED WORD

```
                                   PUT K* LINES ON TO
                                   PROCESSOR BUS
```

(K IS A NUMBER)
(N IS A NUMBER) K<

```
        IF IDLE
```

```
       NO ACTION
```
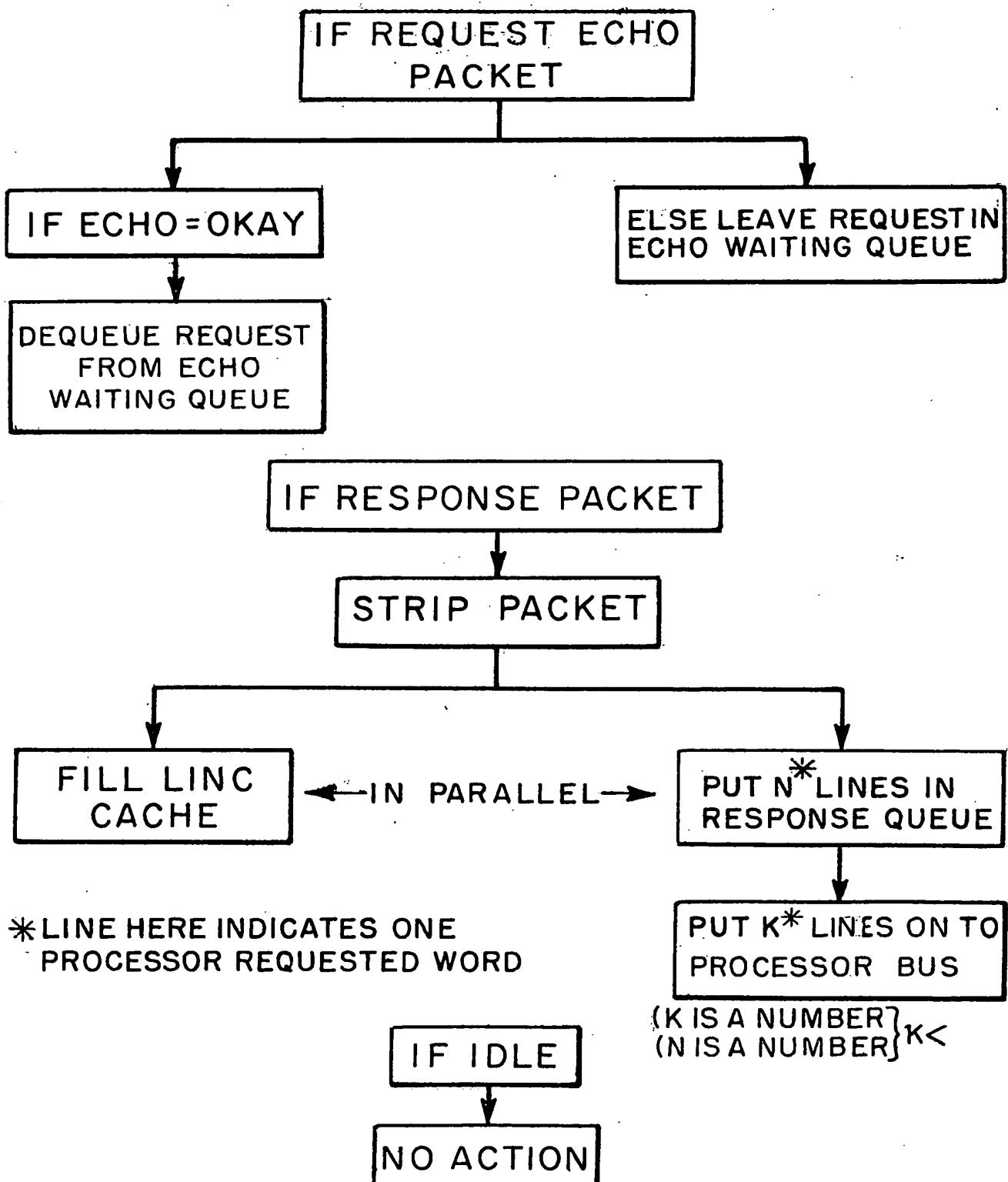
FIG. 28

PROCESSOR NODE TRANSMIT PROTOCOL

ALL ACTIONS WITHIN [ ] HAPPEN IN PARALLEL

CHECK LOAD OR STORE *1

IF STORE

```
PUT IN STORE
ACCUMULATOR
```

```
IF STORE ACCUMULATOR FULL
THEN PACKETIZE & TRANSMIT
```

```
MOVE REQUEST TO *2
ECHO WAITING QUEUE
```

IF LOAD

```
o CHECK STORE ACCUMULATOR
o CHECK LINC CACHE
o ENQUEUE IN REQUEST QUEUE
```

IF IN STORE ACCUMULATOR ②

```
o WRITE BACK TO PROCESSOR
o SET PACKETIZE BIT
```

IF NOT IN
① OR ②

```
PACKETIZE REQUEST BASED
ON PACKETIZE BIT
```

```
TRANSMIT PACKET IF RETRY
IS NOT BEING TRANSMITTED
```

IF
RETRY

```
MOVE REQUEST TO *2
ECHO WAITING QUEUE
```

ELSE (NOT LOAD OR STORE)

```
IF REQUEST QUEUE EMPTY
TRANSMIT IDLE
```

IF NOT IN STORE ACCUMULATOR

BUT IN LINC CACHE ①

```
o WRITE BACK TO PROCESSOR
o SET DEPACKETIZE BIT
o UPDATE HIT COUNTER
```

*1 THE TYPE OF INSTRUCTIONS WILL
BE PROCESSOR SPECIFIC, THERE
MIGHT BE DIFFERENT SUBSETS OF
LOAD & STORE INSTRUCTIONS
WHICH ALSO WILL FOLLOW
THE SAME PROCEDURE.

*2 ITS A DESIGN CHOICE TO PUT
PACKETIZED REQUESTS OR THE RAW
INFORMATION IN THE ECHO
WAITING QUEUE.

FIG. 29

MEMORY NODE RECEIVE PROTOCOL
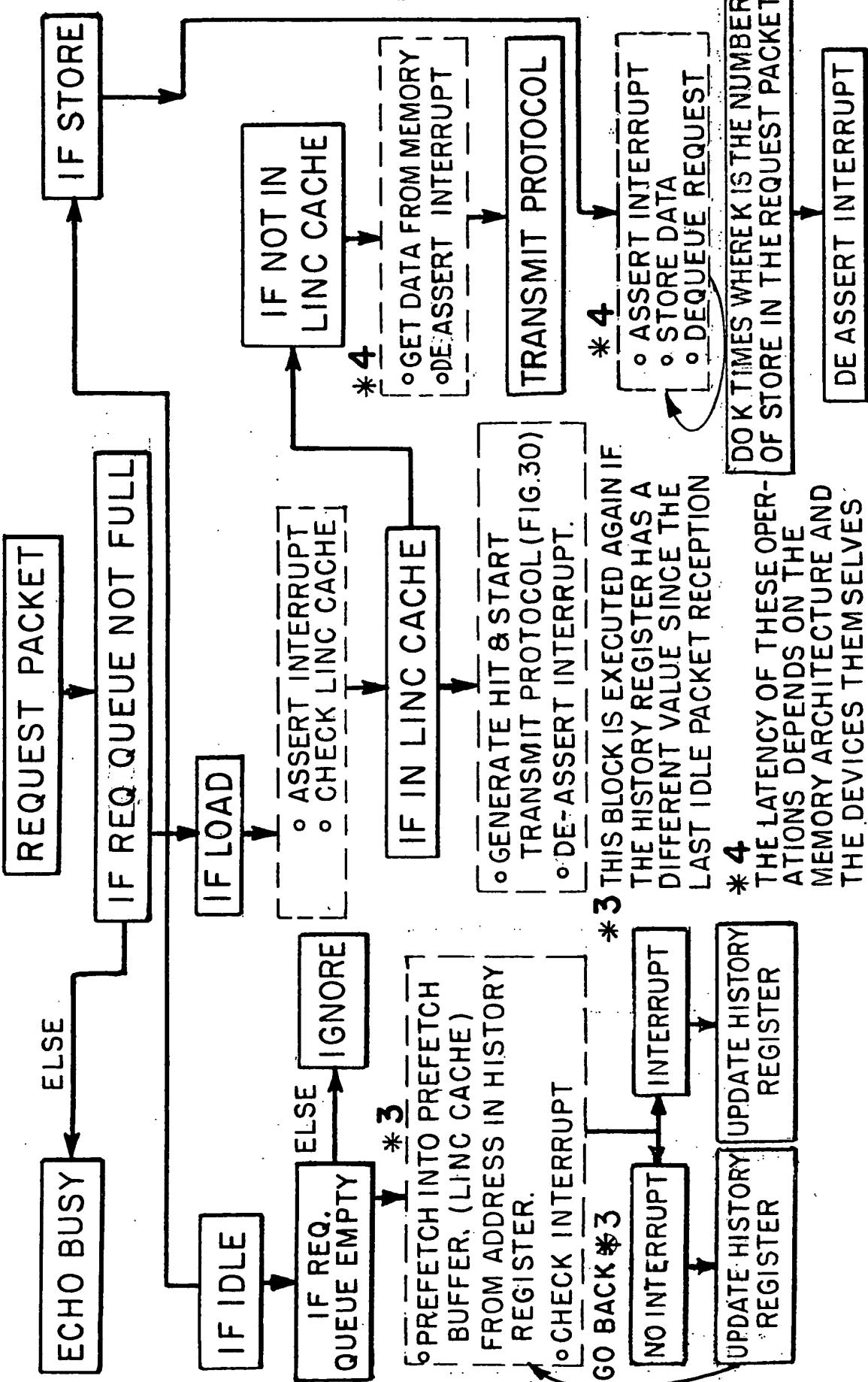
FIG. 30

MEMORY NODE TRANSMIT PROTOCOL

DATA FROM LINC CACHE

DATA FROM MEMORY

PACKETIZE

TRANSMIT

NO DATA

SEND IDLE PACKET